1985

A Factoring Algorithm Using Polygon-to-Chain Reductions for Computing K-Terminal Network Re1 iabil ity

Wood, R. Kevin

A Factoring Algorithm Using Polygon-to-Chain Reductions for Computing *K*-Terminal Network Reliability

R. Kevin Wood
Department of Operations Research, Naval Postgraduate School, Monterey, California 93943

Let $G = (V,E)$ be an undirected graph whose edges may fail, and let G_K denote G with a set $K \subseteq V$ specified. Edge failures are assumed to be statistically independent and to have known probabilities. The K-terminal reliability of G_K, denoted $R(G_K)$, is the probability that all vertices in K are connected by working edges. Computing K-terminal reliability is an NP-hard problem not known to be in NP. A factoring algorithm for computing network reliability recursively applies the formula $R(G_K) = p_i R(G_K{}'*e_i) + q_i R(G_K - e_i)$, where $G_K{}'*e_i$ is G_K with edge e_i contracted, $G_K - e_i$ is G_K with e_i deleted and $p_i = 1 - q_i$ is the reliability of edge e_i. Various reliability-preserving reductions may be performed after each factoring operation in order to reduce computational complexity. The complexity of a slightly restricted factoring algorithm using standard reductions, along with newly developed polygon-to-chain reductions, will be bounded below by an invariant of G, the "minimum domination." For $2 \le |K| \le 5$ or $|V| - 2 \le |K| \le |V|$, this bound is always achievable. The factoring algorithm with polygon-to-chain reductions will always perform as well as or better than an algorithm using only standard reductions, and for some networks, it will outperform the simpler algorithm by an exponential factor. This generalizes early results that were only valid for $K = V$. Removing the restriction on edge selection leaves results essentially unchanged in the upper range of $|K|$, but minimum domination becomes only a tight upper bound for the lower range.

1. INTRODUCTION

Analysis of network reliability is important in computer, communication, power, and various other networks. Components of a particular network may be subject to random failure and the network may or may not continue to function after some of its components have failed. We wish to determine, as efficiently as possible, the probability that the network is functional. The purpose of this article is to develop and analyze the complexity of a general factoring-based algorithm for exact computation of network reliability.

The network model used in this paper may be thought of as a communication network

NETWORKS, Vol. 15 (1985) 173–190

CCC 0028-3045/85/020173-18$04.00

with duplex communication links connecting various transceiving stations. Communication can pass in both directions along a link if the link is working. No communication in either direction is possible if the link has failed. The network is considered functional if a specified set of the transceiving stations is able to communicate.

The formal definition of the *K-terminal network reliability problem* (also known as the "Steiner network reliability problem") is as follows: Let $G = (V,E)$ be a graph whose edges may fail independently of each other, with known probabilities. Every vertex $v_j \in V$ is assumed to be perfectly reliable. The *edge-failure probability* for edge $e_i \in E$ is given by q_i and the *edge reliability* is given by $p_i = 1 - q_i$. Now, a set $K \subseteq V$ must be specified for G. These vertices will be referred to as the *K-vertices* of G, and G_K will be used to denote the graph G with K specified. The *K-terminal reliability* of G_K, denoted $R(G_K)$, is the probability that all K-vertices in G_K are connected by working edges where $R(G_K) \equiv 1$ if $|K| = 1$. (We will use the shorter phrase "G_K is connected" to mean that the K-vertices in G_K are connected.) For historical reasons, most authors (see Hwang *et al.* [9] for a review and bibliography) have considered the computation of $R(G_K)$ only when $|K| = 2$ or $K = V$. However, the case in which $2 \leq |K| \leq |V|$ has been handled, in varying degrees of generality, by some authors (Ball [1], Buzacott [4], Johnson [10], Rosenthal [14], Satyanarayana [15], Wood [22]).

The K-terminal network reliability problem belongs to the class of NP-hard problems not known to be in NP (Ball [1], Ball and Provan [2], Rosenthal [13], Valiant [21]). It follows from Valiant [21] that the K-terminal problem also belongs to the class of #P-complete (number P-complete) problems which are equivalent to counting (not listing) the number of solutions to an NP-complete problem. Special cases of the K-terminal reliability problem, the two-terminal and all-terminal problems, have been shown to be #P-complete by Valiant [21] and by Ball and Provan [2], respectively. Although the K-terminal reliability problem is theoretically intractable, the situation is not hopeless. For practical problems involving sparse networks, fairly large networks can be analyzed, and techniques developed in this article extend the range of problems for which reliability can be exactly computed.

The factoring theorem of network reliability is of primary importance in this paper. This theorem establishes the validity of the following conditional reliability formula:

$$R(G_K) = p_i R(G_{K'} * e_i) + q_i R(G_K - e_i),$$

where $G_{K'} * e_i$ is G_K with edge e_i contracted, K' is K suitably redefined if either endpoint of e_i is in K, and $G_K - e_i$ is G_K with edge e_i deleted. This will be more formally defined in Section 3. The factoring theorem can be used to establish reliability-preserving reductions which reduce G_K to a smaller graph $G'_{K'}$ such that $R(G_K) = \Omega R(G'_{K'})$, where Ω is a constant resulting from the reduction. The factoring theorem is also the basis for a whole class of algorithms for computing network reliability. Moskowitz [12] was the first to employ the factoring theorem directly as a means of calculating network reliability. The above equation can be recursively applied to the induced graphs and reliability-preserving reductions made where applicable within the recursion. Eventually the induced graphs are reduced to simple structures, like single edges, for which reliability is trivially computed, or some K-vertices become discon-

nected, in which case the reliability of the induced graph is zero. In this way, the reliability of any network may be computed, at least in theory. This method of computing network reliability is known as *factoring* and is a special case of pivotal decomposition of a binary coherent system (Barlow and Proschan [3]). Other exact methods for network reliability evaluation do exist, including inclusion–exclusion methods (e.g., Satyanarayana and Prabhakar [17], Satyanarayana [15]), "composition" (Buzacott [4]), and boolean-algebra methods (e.g., Fratta and Montanari [6]). However, factoring methods seem to be the most promising except for very dense graphs and graphs with special topologies. Only factoring-based algorithms and their complexity will be discussed here.

In this paper, we analyze the complexity of a factoring algorithm which employs a new set of reliability-preserving reductions. Until recently, the only reliability-preserving reductions that were usually included within a factoring algorithm were the well-known series and parallel reductions, the degree-two reduction which is an extension of the series reduction (Rosenthal [13]), and the bridge contraction (e.g., Johnson [10]). With few exceptions, analysis of factoring algorithms has essentially been limited to algorithms using only these reductions. However, a new set of polygon-to-chain reductions has been developed (Satyanarayana and Wood [18]) which is particularly useful in the K-terminal problem. These reductions replace parallel chains of edges with single chains. Satyanarayana and Wood also give an $O(|E|)$ algorithm that performs all series, parallel, degree-two, and polygon-to-chain reductions on a general network so the new reductions can be implemented efficiently.

Satyanarayana and Chang [16] have used graph invariants to analyze the complexity of the K-terminal reliability problem using the factoring algorithm along with series and parallel reductions. They relate a graph invariant called "domination," denoted $D(G_K)$, to the backtrack search structure produced by the factoring algorithm. Domination has a factoring theorem associated with it, and using this theorem they show that the factoring algorithm will be optimal, i.e., have the minimum possible number of leaf nodes in its backtrack search structure, if that number is equal to $D(G_K)$. Furthermore, they show that this will be true if and only if a particular edge-selection strategy is used for factoring. Chang [5] uses "minimum domination," $\mu(G) = \min_{K:|K|=2} D(G_K)$, to find an optimal factoring algorithm for the all-terminal problem which uses degree-two and parallel reductions. Johnson [10] shows that minimum domination is equivalent the Crapo beta-invariant of the graphic matroid.

We first generalize Chang's results on the all-terminal problem to the K-terminal problem when $|K|$ is within certain limits. Using a restricted edge-selection strategy, we show that for $2 \leq |K| \leq 5$ or for $|V| - 2 \leq |K| \leq |V|$, it is always possible to compute $R(G_K)$ in time which is proportional to $\mu(G)$. Although not completely general, this complexity result is significant since domination may be exponentially larger than minimum domination, and only domination results were previously known for the K-terminal problem. It also means that we can compute the most common measure of reliability, two-terminal reliability, in approximately the same amount of time for any two terminals in a given graph. Finally, we remove the restriction on the edge-selection strategy and show that $\mu(G)$ provides a tight upper bound on algorithmic complexity.

The rest of this paper is outlined as follows: In Section 2, we define necessary graph-theoretic terms, and in Section 3, we describe the reliability-preserving reductions to be used in the factoring algorithm. In Section 4, we formally describe the factoring theorem of network reliability. In Section 5, we define an algorithmic framework for the factoring algorithm, describe this algorithm's binary search structure and review earlier results using graph invariants for complexity analysis. In Section 6 we prove the new results on the K-terminal network reliability problem. Section 7 provides a brief conclusion and suggests additional techniques for devising even more efficient factoring algorithms.

2. GRAPH-THEORETIC DEFINITIONS

In this section, we define a few basic graph-theoretic terms and emphasize certain concepts that are useful in this paper. A *graph* $G = (V,E)$ is composed of two finite sets: V is the set of *vertices* and E is the set of *edges*. Each edge $e \in E$ corresponds to an unordered pair of vertices, that is, $e = (u,v)$, where $u,v \in V$. The vertices u and v are called the *endpoints* of edge e, and e is said to be *incident* to u and v. In the definition of graph used in this paper, we will allow *parallel* edges, i.e., multiple edges with the same endpoints and *self-loops*, i.e., edges of the form $e = (u,u)$. The *degree* of a vertex, denoted "deg(v)," is the number of edges incident to v except that self-loops are counted twice.

Two vertices u and v are *connected* if there exists a sequence of vertices and edges of the form u, (u,v_1), v_1, (v_1,v_2), . . . , (v_{l-1},v_l), v_l, (v_l,v), v. Such a sequence is a *path*, and the path is a *cycle* if the first and last vertices are identical. A set of vertices $K \subseteq V$ is connected if there exists a path between all pairs of vertices in K. G is said to be connected if V is connected.

A graph $G' = (V',E')$ is a *subgraph* of $G = (V,E)$ if $V' \subseteq V$ and $E' \subseteq E$. Let V_0 be a subset of the vertices of V in G and let $G - V_0$ be the subgraph of G obtained from G by deleting all vertices $v \in V_0$ and all edges incident to those vertices. Let V_0 be a smallest set of vertices such that $G - V_0$ is disconnected or $|V - V_0| = 1$. G is said to be *k-connected* if $|V_0| \geq k$ (Tutte [20]). We will use the standard terms "biconnected" and "triconnected" to mean 2-connected and 3-connected, respectively. If G is connected but $G - v$ is disconnected, then v is a *cutvertex* of G. A connected graph is *nonseparable* if it contains no cutvertices. Note that a nonseparable graph is either a single vertex, a single edge with its two end vertices or it is biconnected.

If a graph G can be partitioned into two components such that $G = G^1 \cup G^2$, $E^1 \cap E^2 = \emptyset$, $V^1 \cap V^2 = \{u,v\}$, $|E^1| \geq 2$ and $|E^2| \geq 2$, then $\{u,v\}$ is a *separating pair*. Letting $e^1 = (u,v)$ and $e^2 = (u,v)$ be two *virtual* or artificial edges, $G^1 + e^2$ and $G^2 + e^2$ are *split components* of G. G may be recursively split until no more splitting is possible. The resulting split components are not necessarily unique, but if all cycles and triple-bonds (three edges in parallel) are merged, then the remaining components will be the unique *triconnected components* of G (Hopcroft and Tarjan [8]).

We next define "series-parallel graph." In a graph, edges with the same end vertices are *parallel edges*. Two nonparallel edges are *adjacent* if they are incident to a common vertex. Two adjacent edges are *series edges* if their common vertex is of degree 2.

Replacing a pair of series (parallel) edges by a single edge is called a *series (parallel) replacement.* A *series-parallel graph* is a graph that can be reduced to a single edge by successive series and parallel replacements.

We define a *chain* χ in a graph to be an alternating sequence of distinct vertices and edges, v_1, (v_1,v_2), v_2, (v_2,v_3), v_3, . . . , v_{k-1}, (v_{k-1},v_k), v_k, such that the internal vertices, v_2, v_3, . . . , v_{k-1}, are all of degree 2 and the end vertices, v_1 and v_k, are of degree greater than 2. A chain need not contain any internal vertices, but it must contain at least one edge and its two end vertices. The length of a chain is simply the number of edges it contains. If two chains χ_1 and χ_2 have common end vertices u and v, i.e., the chains are in parallel, then $\chi_1 \cup \chi_2$ is a *polygon.* Two parallel edges constitute a polygon but we will normally refer to such polygons as parallel edges.

The reader should consult a standard text such as Harary [7] for a more basic and comprehensive discussion of graph theory and for any terms not defined here.

3. RELIABILITY-PRESERVING REDUCTIONS

In order to reduce the size of graph G_K and therefore reduce the complexity of computing $R(G_K)$, *reliability-preserving reductions* can be applied. These reductions alter a subgraph of G_K topologically and probabilistically to obtain $G'_{K'}$ such that $R(G_K) = \Omega R(G'_{K'})$. Ω is a multiplicative constant derived exclusively from the original subgraph. The following three reductions are usually referred to as "simple" or "standard" reductions.

(*R*1) Let $e_a = (u,v)$ and $e_b = (u,v)$ be two parallel edges in G_K. A *parallel reduction* obtains G' by replacing e_a and e_b with a single edge $e_c = (u,v)$ such that $p_c = 1 - q_a q_b$, and it defines $\Omega = 1$ and $K' = K$.

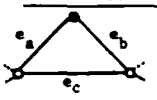
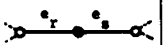
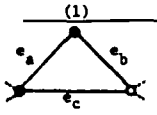
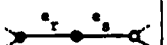
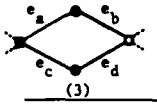
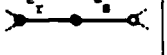
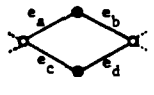
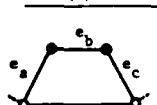
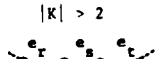
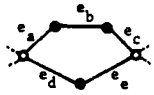
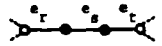
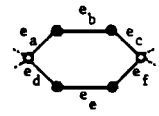
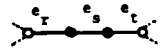
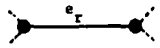
(*R*2) Let $e_a = (u,v)$ and $e_b = (u,w)$ be two series edges in G_K such that $\deg(v) = 2$ and $v \notin K$. A *series reduction* obtains G' by replacing e_a and e_b with a single edge $e_c = (u,w)$ such that $p_c = p_a p_b$, and it defines $\Omega = 1$ and $K' = K$.

(*R*3) Let $e_a = (u,v)$ and $e_b = (u,w)$ be two series edges in G_K such that $\deg(v) = 2$ and $u,v,w \in K$. A *degree-two reduction* obtains G' by replacing e_a and e_b with a single edge $e_c = (u,w)$ such that $p_c = p_a p_b/(1 - q_a q_b)$, and it defines $\Omega = (1 - q_a q_b)$ and $K' = K - v$.

Another set of reductions has recently been introduced which replaces a polygon with a chain (Satyanarayana and Wood [18]). The main point of this paper is to show that these polygon-to-chain reductions can significantly reduce the work required by a factoring algorithm. Consider a graph G_K which does not admit any simple reductions but does contain some polygon. In general, no such polygon need exist, but, if it does exist, then the number of possible configurations is limited. This follows from the facts that, after all simple reductions have been made, (i) every degree-two vertex of G_K is a K-vertex, (ii) there can be no more than two K-vertices in a chain, and (iii) the length of any chain in G_K is at most 3. Table 1 shows the possible polygons and associated reductions.

(*R*4) Let Δ in G_K be a type i polygon as shown in row i of Table 1. A *polygon-to-chain* reduction obtains G' by replacing Δ with the corresponding chain χ and it defines Ω and K', as shown in row i of Table 1.

TABLE I. Polygon-to-chain reductions.

Polygon Type	Chain Type	Reduction Formulas	New Edge Reliabilities
 (1)		$\alpha = q_a p_b q_c$ $\beta = p_a q_b q_c$ $\delta = p_a p_b p_c \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c}\right)$	$p_r = \frac{\delta}{\alpha + \delta}$ $p_s = \frac{\delta}{\beta + \delta}$ $\Omega = \frac{(\alpha + \delta)(\beta + \delta)}{\delta}$
 (2)		$\alpha = q_a p_b q_c$ $\beta = p_a q_b q_c$ $\delta = p_a p_b p_c \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c}\right)$	
 (3)		$\alpha = p_a q_b q_c p_d + q_a p_b p_c q_d + q_a p_b q_c p_d$ $\beta = p_a q_b p_c q_d$ $\delta = p_a p_b p_c p_d \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	
 (4)		$\alpha = q_a p_b q_c p_d$ $\beta = p_a q_b q_c p_d + q_a p_b p_c q_d$ $\delta = p_a q_b p_c q_d$ $\gamma = p_a p_b p_c p_d \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	
 (5)	$\lvert K\rvert > 2$  See note	$\alpha = q_a p_b p_c q_d$ $\beta = p_a q_b p_c q_d$ $\delta = p_a p_b q_c q_d$ $\gamma = p_a p_b p_c p_d \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d}\right)$	$p_r = \frac{\gamma}{\alpha + \gamma}$ $p_s = \frac{\gamma}{\beta + \gamma}$ $p_t = \frac{\gamma}{\delta + \gamma}$ $\Omega = \frac{(\alpha + \gamma)(\beta + \gamma)(\delta + \gamma)}{\gamma^2}$
 (6)		$\alpha = q_a p_b p_c q_d p_e$ $\beta = p_a q_b p_c (p_d q_e + q_d p_e) + p_b (q_a p_c p_d q_e + p_a q_c q_d p_e)$ $\delta = p_a p_b q_c p_d q_e$ $\gamma = p_a p_b p_c p_d p_e \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e}\right)$	
 (7)		$\alpha = q_a p_b p_c q_d p_e p_f$ $\beta = p_a q_b p_c (q_d p_e p_f + p_d q_e p_f + p_d p_e q_f) + p_a p_b q_c p_f (p_d q_e + q_d p_e) + q_a p_b p_c p_d (q_e p_f + p_e q_f)$ $\delta = p_a p_b q_c p_d p_e q_f$ $\gamma = p_a p_b p_c p_d p_e p_f \left(1 + \frac{q_a}{p_a} + \frac{q_b}{p_b} + \frac{q_c}{p_c} + \frac{q_d}{p_d} + \frac{q_e}{p_e} + \frac{q_f}{p_f}\right)$	Note: For $\lvert K\rvert = 2$, new chain is  $p_r = (p_b + p_a q_b p_c p_d)/\Omega$ $\Omega = p_b + p_a q_b p_c$

Note: Darkened vertices represent K-vertices.

4. THE FACTORING THEOREM OF NETWORK RELIABILITY

Let $e_i = (u,v)$ be some edge of a graph G_K, let F_i denote the event that e_i is working, and let $\overline{F_i}$ denote the complementary event. Since $R(G_K)$ is just a probability, the rules of conditional probability can be applied to obtain

$$R(G_K) = p_i R(G_K|F_i) + q_i R(G_K|\overline{F}_i). \tag{1}$$

$G_K|F_i$ actually defines a new graph in which u and v are known to be connected. This new *induced* graph (we do not mean induced subgraph in the standard graph theoretic sense) denoted $G_{K'}*e_i$, is obtained by deleting e_i and merging u and v into a single supervertex $w = u \cup v$. If either u or v is a K-vertex, then w is a K'-vertex. More formally, $G_{K'}*e_i$ is defined by

$$G*e_i = (V - u - v + w, E - e_i), \qquad w = u \cup v,$$
$$K' = \begin{cases} K & \text{if } u,v \notin K \\ K - u - v + w & \text{if } u \in K \text{ or } v \in K. \end{cases} \tag{2}$$

Similarly, $G_K|\bar{F}_i$ defines a new graph denoted $G_K - e_i$, where $G - e_i = (V, E - e_i)$. Figure 1 illustrates how these two graphs are induced. We can now write Equation (1) as

$$R(G_K) = p_i R(G_{K'}*e_i) + q_i R(G_K - e_i). \tag{3}$$

The validity of this relationship was first shown by Moore and Shannon [11] and is known as the factoring theorem for network reliability.

This factoring theorem is useful in two ways. First, it can be used as a method to derive and prove the validity of the reliability-preserving reductions already described. Second, the factoring theorem is the basis for a whole class of algorithms for computing network reliability. Moskowitz [12] was the first to employ the factoring theorem directly as a means of calculating network reliability. Equation (3) can be recursively applied to the induced graphs and reliability-preserving reductions can be made where applicable within the recursion. Eventually the induced graphs are reduced to simple structures, like single edges, for which reliability is trivially computed, or some K-vertices become disconnected, in which case the reliability of the induced graph is zero. In this way, the reliability of any network may be computed, at least in theory. This method of computing network reliability is known as *factoring* and is a special case of pivotal decomposition of a binary coherent system (Barlow and Proschan [3]).

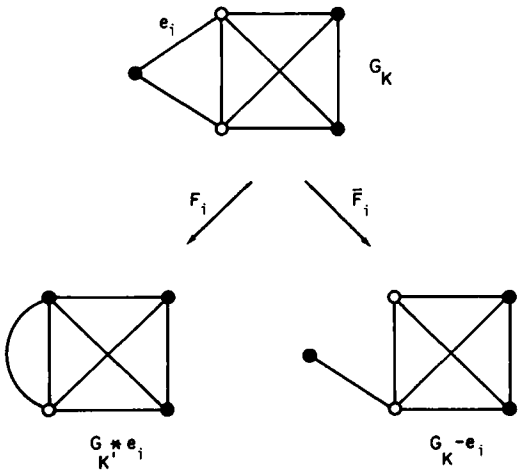


FIG. 1. Induced graphs obtained by factoring on e_i.

5. A FACTORING ALGORITHM

The following algorithm, which employs the recursive function REL (this name is not original to this article!), describes a general framework for exact computation of network reliability via factoring. The algorithm employs a set of reductions denoted **R** and an edge-selection strategy denoted **S**. An edge-selection strategy is a set of rules which indicates which edges may be selected for factoring. Such a strategy is well-defined only if, at every stage of the factoring algorithm, there exists at least one edge which satisfies the specified rules. Variations on this algorithm are possible, but the algorithm given will be sufficient for our purposes. We assume that the input graph G_K is connected and nonseparable, since otherwise its components can be easily identified and $R(G_K)$ computed by evaluating the components separately (Rosenthal [13]).

Algorithm 1

MAIN
Input: A nonseparable graph G_K with associated edge probabilities.
Output: K-terminal reliability of G_K.

Begin
  $R \leftarrow REL(G_K)$.
  Print(''$R(G_K)$ *is*'' R).
  Stop.
End *of MAIN*

Function $REL(G_K)$

Input: Graph G_K with associated edge reliabilities.
Output: Returns the value R which is the K-terminal reliability of G_K.

Begin
  $M \leftarrow 1$.
  If G_K *is disconnected* **then** *Return*(0).
  If $|K| = 1$ **then** *Return* (1).
  Delete any isolated vertices from G_K.
  Until *no reliability-preserving reductions can be made* **do**
    Begin
      Perform some reduction from **R** *on* G_K *to obtain* $G'_{K'}$ *and* Ω *such that* $R(G_K) = \Omega R(G'_{K'})$.
      $M \leftarrow M\Omega$.
      $G_K \leftarrow G'_{K'}$.
    End
  If G_K *is single edge* e_i **then** *Return*(Mp_i).
  Select an edge e_i *using strategy* **S**.
  $R \leftarrow M(p_i REL(G_{K'} * e_i) + q_i REL(G_K - e_i))$.
  Return(R).
End *of REL*.

The exact complexity of the above algorithm will, of course, depend on what sort of reductions are used and how much work is required to select an edge for factoring. However, since the number of calls to REL is generally exponential, while the reductions and edge-selection strategies will be of polynomial complexity, we can use the number of calls to REL as a measure of algorithmic complexity. Each call to REL corresponds to a *node* of the related *binary backtrack search structure*. A *leaf node* (or simply "leaf") is a node with no further nodes below it. Let $N(G_K)$ and $L(G_K)$ denote, respectively, the number of nodes and the number of leaves in the search structure associated with Algorithm 1 applied to G_K. Then, $N(G_K) = 2L(G_K) - 1$ since the search structure is binary. Thus, we can and will use $L(G_K)$ as the measure of the complexity of Algorithm 1. An edge-selection strategy **S** for Algorithm 1 is said to be *optimal* with respect to reductions **R** if $L(G_K)$ is minimized by **S**.

In the next section, we consider $\mathbf{R} = \{R1,R2,R3,R4\}$, and to show a generalization of earlier results, restrict our search for an optimal edge-selection strategy to $\mathbf{S} \subseteq \mathbf{S}_1$, where $\mathbf{S}_1$ specifies that no induced graph may have $|K| = 1$. With this restriction, we find that the graph invariant "minimum domination" (Chang [5]) provides a lower bound on $L(G_K)$ and prove that this lower bound can always be achieved when $2 \leq |K| \leq 5$ or when $|V| - 2 \leq |K| \leq |V|$. By then removing the restriction that **S** belong to $\mathbf{S}_1$, we find that $L(G_K)$ is essentially unchanged for $|V| - 2 \leq |K| \leq |V|$, but may actually be less than the value of the invariant for $2 \leq |K| \leq |5|$; thus, even though the optimal edge-selection strategy cannot be determined, we have an algorithm of bounded complexity which is demonstrably better than previously available algorithms. In order to carry out this analysis, we next discuss two graph invariants and show how one of them has been used to analyze the all-terminal reliability problem. The definitions given next are from Satyanarayana [15].

A *K-tree* (or "Steiner tree") of a graph G_K is any minimal graph that connects all the K-vertices of G_K. An edge is *irrelevant* if it does not occur in any K-tree. A *formation* of G_K is a set of K-trees whose union is G_K. Note that G_K has no formations if it contains any irrelevant edges. Letting N_0 be the number of odd cardinality formations of G_K and letting N_e be the number of even cardinality formations of G_K, then the *domination* of G_K is defined by

$$D(G_K) = |N_0 - N_e|.$$

Clearly, $D(G_K)$ is a combinatorial invariant of G_K. Using a factoring theorem associated with domination, Satyanarayana and Chang [16] show how Algorithm 1 with $\mathbf{R} = \{R1,R2\}$ must have $L(G_K) \geq D(G_K)$. If the algorithm never factors so as to create graphs with irrelevant edges or graphs that are disconnected, then $L(G_K) = D(G_K)$ and the algorithm will be optimal. An edge-selection strategy which achieves this result is easily implemented. We will not go into further detail about this work, but rather describe a different graph invariant which will be used in this paper. This invariant is called *minimum domination* and is defined by

$$\mu(G) = \min_{K:|K|=2} D(G_K).$$

From its definition, it follows that $\mu(G)$ is a nonnegative integer for any graph G.

Additional properties of $\mu(G)$ are established in Chang [5]:

Property 1. (a) $\mu(G) = \mu(G - e) + \mu(G*e)$.
(b) $\mu(G)$ is invariant under series and parallel replacements if G is biconnected.
(c) $\mu(G) > 0$ if and only if G has no self-loops and is biconnected.
(d) $\mu(G) = 1$ if and only if G is a single edge or a biconnected series-parallel graph.

Property 1(a) is a factoring theorem for minimum domination, and along with the other properties, it will enable us to analyze $L(G_K)$ for Algorithm 1. To complete the relationship between minimum domination and the factoring algorithm for network reliability, we need the following property which extends property 1(b) to the domain of reliability-preserving reductions.

Property 2. Let $G'_{K'}$ be obtained from a biconnected graph G_K by any $R1$–$R4$ reduction. Then, $\mu(G') = \mu(G)$.

The property is obvious for $R1$, $R2$, and $R3$ since these correspond directly to parallel or series replacements. Topologically, a polygon-to-chain reduction can be implemented as sequence of one or more series replacements, a parallel replacement, and one or more inverse series replacements. Thus, the property is easily seen to be true for $R4$, also.

Under certain conditions, $L(G_K) = \mu(G)$ and this is the best that can be achieved. Unfortunately, there is no known way of computing $\mu(G)$ other than using a factoring algorithm or other exponentially complex procedure unless G possesses some special, symmetric structure. So, while the complexity results enable us to find an optimal algorithm, they do not normally yield any *a priori* estimate on the computational requirements of a specific problem.

Chang [5] first shows how an algorithm can be specified for the all-terminal problem which will optimally produce only $\mu(G)$ leaves. We will not reproduce his results directly, but rather generalize immediately to the K-terminal problem and then show that his results follow as a special case.

6. OPTIMAL EDGE-SELECTION FOR THE *K*-TERMINAL PROBLEM

To simplify the initial discussion, the set of admissible edge-selection strategies will be restricted to:

$\mathbf{S_1}$ Any edge in G_K may be selected except an edge e such that $|K'| = 1$ in $G_{K'}*e$.

This restriction is necessary for optimality of Algorithm 1 under $\mathbf{R} = \{R1,R2\}$ (Satyanarayana and Chang [16]). As will be seen later, however, relaxation of this restriction can lead to decreases in $L(G_K)$ for Algorithm 1 under $\mathbf{R} = \{R1,R2,R3,R4\}$.

We first establish that $\mu(G)$ is a lower bound on $L(G_K)$ for Algorithm 1 under $\mathbf{S} = \mathbf{S_1}$ and $\mathbf{R} = \{R1,R2,R3,R4\}$. Then, under the same conditions we show that an optimal edge-selection strategy can be specified when $2 \leq |K| \leq |5|$ or $|V| - 2 \leq |K| \leq |V|$. That is, we show that $L(G_K) = \mu(G)$ can always be obtained for K in the specified ranges. A simple, linear-time edge-selection strategy, a subset of $\mathbf{S_1}$, will achieve this

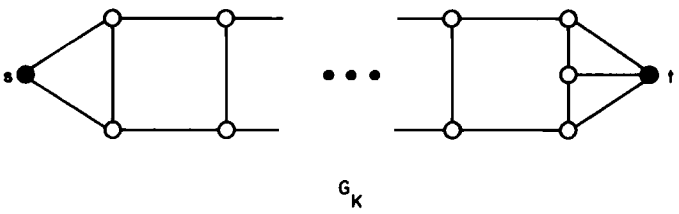


FIG. 2. Graph with $D(G_K) = 3 \times 2^{(|E|-7)/3}$ and $\mu(G) = 2$.

result. These results, while not completely general, are significant from a computational point of view. Consider the graph of Figure 2. If we use the factoring algorithm of Satyanarayana and Chang [16] that employs $\mathbf{R} = \{R1,R2\}$, the optimal algorithm has $L(G_K) = D(G_K) = 3 \times 2^{(|E|-7)/3}$. With the addition of $R3$ and $R4$ reductions, the optimal algorithm under $\mathbf{S}_1$ has $L(G_K) = \mu(G) = 2$.

A Lower Bound on $L(G_K)$ Under S_1

Theorem 1. Let G_K be a biconnected graph with no self-loops and with $|K| \geq 2$. Then, for a factoring algorithm with $\mathbf{R} = \{R1,R2,R3,R4\}$ and $\mathbf{S} \subseteq \mathbf{S}_1$, $\mu(G) \leq L(G_K)$, i.e., $\mu(G)$ is a lower bound on the number of leaves created by the factoring algorithm.

Proof. Let L be the set of leaf node graphs produced by the algorithm. L will consist of two disjoint sets: L_1, those leaf node graphs which are single edges and all of whose proper ancestor graphs are biconnected; and L_2, those leaf node graphs which have a proper ancestor which is not biconnected. These are the only possibilities since, as specified, Algorithm 1 will create a leaf node only when G_K consists of single edge or G_K is disconnected, and, if G_K is disconnected, G must have a proper ancestor which is not biconnected. Now define N_2 to be all those graphs in the search structure which are separable but whose proper ancestors are all biconnected. Every $G' \in L_2$ has an ancestor in N_2. Thus, $|N_2| \leq |L_2|$. Now, L_1 and N_2 taken with their ancestors form a binary structure such that

$$\begin{aligned}
\mu(G) &= \sum_{G' \in L_1 \cup N_2} \mu(G') \text{ by properties 1(a) and 2} \\
&= \sum_{G' \in L_1} \mu(G') + \sum_{G' \in N_2} \mu(G') \\
&= \sum_{G' \in L_1} 1 + \sum_{G' \in N_2} 0 \text{ by properties 1(c) and 1(d)} \\
&= |L_1|.
\end{aligned}$$

Thus, $\mu(G) = |L_1| \leq |L| = L(G_K)$. ∎

Corollary. Let G_K be a biconnected graph with no self-loops and with $|K| \geq 2$. If an edge e can be selected at each factoring step of Algorithm 1 (under $\mathbf{S}_1$ and

$\mathbf{R} = \{R1,R2,R3,R4\}$) such that $G - e$ and G^*e are both biconnected, then that edge-selection strategy is optimal.

Proof. Under the specified conditions, $N_2 = \emptyset$, so $L(G_K) = |L| = |L_1| = \mu(G)$. ∎

Consider how the above results can be applied to the all-terminal problem. First, note that $\mathbf{S}_1$ will always be satisfied. Second, note that only $R1$ and $R3$ are relevant reductions and that performing all such reductions ensures that no series or parallel replacements remain in G. Chang [5] shows that for any biconnected graph G which admits no series or parallel replacements there always exists an edge e such that $G - e$ and G^*e are biconnected. If such an edge can be identified, then $L(G_V) = \mu(G)$ by the corollary and the algorithm is optimal. Chang determined that an optimal edge-selection strategy is to select any edge that is not incident to any vertex of a separating pair. If G is triconnected, any edge will do. Otherwise, an appropriate edge can be identified by examining the triconnected components and separating pairs of G, and therefore, the edge-selection can be carried out in linear time (Hopcroft and Tarjan [8]). Johnson gives a less restrictive strategy by showing that (i) if G is biconnected, there must exist a triconnected split component of G which has only one virtual edge and at least five real edges, and (ii) any real edge of such a component may be selected for factoring.

Optimal Factoring under $\mathbf{S}_1$

Here we show that an optimal edge-selection strategy can be specified for Algorithm 1 under $\mathbf{R} = \{R1,R2,R3,R4\}$ and $\mathbf{S} \subseteq \mathbf{S}_1$ when $2 \leq |K| \leq 5$ or $|V| - 2 \leq |K| \leq |V|$. Figure 3 is used to show why the results are limited to such unusual values of $|K|$. As in Chang [5], the optimal algorithm under $\mathbf{S}_1$ will depend on always being able to find an edge of G on which to factor such that both G^*e and $G - e$ are biconnected. Figures 3a and 3b show minimal irreducible graphs with $|K| = 6$ and $|K| = |V| - 3$, respectively. No $R1$–$R4$ reductions are possible. Note that no matter which edge is selected for factoring, $G - e$ must be separable.

The following lemma is necessary for proving that, under $\mathbf{S}_1$, it is always possible to find a suitable edge for factoring if $|K|$ is within the given ranges.

FIG. 3. Minimal irreducible graphs.

Lemma 1. Let G be a biconnected graph which has no series or parallel edges but which is not triconnected. Then, there exist two triconnected split components of G, each containing at least five real edges and only one virtual edge.

Proof. Since G is biconnected but not triconnected, it must contain at least one separating pair. Let $\{u,v\}$ be the separating pair which allows us to partition G into two graphs G^1 and G^2 with $G = G^1 \cup G^2$, $E^1 \cap E^2 = \emptyset$, $V^1 \cap V^2 = \{u,v\}$, $|E^1| \geq 2$, $|E^2| \geq 2$, and such that E^1 is minimal. Letting $e^1 = (u,v)$ be a virtual edge, $G^1 + e^1$ is a triconnected split component of G since it cannot be split any further. Since G had no series or parallel edges, $|V^1| \geq 4$ and $|E^1| \geq 5$ (Satyanarayana and Chang [16]).

Next, consider $G^2 + e^2$, where $e^2 = (u,v)$ is a virtual edge. If $G^2 + e^2$ is triconnected, it is the second split component for which we were looking and we are done. Otherwise, $G^2 + e^2$ must be biconnected and contain at least one separating pair. Partition $G^2 + e^2$ by a separating pair $\{x,y\}$ such that $G^2 + e^2 = G^3 \cup G^4$, etc., but $e^2 \notin E^3$, no edge parallel to e^2 is in E^3, and E^3 is minimal. Since $e^2 \notin E^3$, G^3 contains no parallel edges and, by construction, it contains no series edges. Thus, as above, $|V^3| \geq 4$ and $|E^3| \geq 5$. Letting $e^3 = (x,y)$, be a virtual edge, $G^3 + e^3$ is a second triconnected split component of G as required since it cannot be split any further. ■

Johnson [10] proves this next lemma.

Lemma 2. If G has no series or parallel edges, no self-loops, and is biconnected but not triconnected, then G^*e and $G - e$ will both be biconnected and contain no self-loops if e is any real edge of a triconnected component of G containing only one virtual edge as specified in Lemma 1.

Consider the following edge-selection strategy.

$\mathbf{S_2}$ Select any edge $e \in E$ such that both G^*e and $G - e$ have no self-loops and are biconnected.

If G is biconnected and without self-loops, by Properties 1(a) and 1(c), $\mathbf{S_2}$ is equivalent to selecting any edge $e \in E$ such that $\mu(G^*e) > 0$ and $\mu(G - e) > 0$. However, edge selection in a factoring algorithm will be carried out in terms of something which is directly observable, in this case, connectivity. We have therefore chosen to state the strategy in terms of connectivity. Our new complexity results for the K-terminal reliability problem will follow directly from the next theorem.

Theorem 2. Let G_K be a nontrivial, biconnected graph with no self-loops and which admits no $R1$–$R4$ reductions. If $2 \leq |K| \leq 5$ or $|V| - 2 \leq |K| \leq |V|$, there exists an edge $e \in E$ satisfying $\mathbf{S_1} \cap \mathbf{S_2}$.

Proof. G_K being nontrivial, biconnected, having no self-loops, and admitting no reductions implies that $\mu(G) > 1$. Thus, Property 1(c) implies that e satisfies $\mathbf{S_2}$ if and only if $\mu(G^*e) > 0$ and $\mu(G - e) > 0$. In the rest of the proof, we let G^+ be G with all chains of length two and three replaced by single edges, and we let

$K^+ = \{v: v \in K, \deg(v) > 2\}$. G^+ will contain no parallel edges since no parallel or polygon-to-chain reductions remain in G_K.

Case 1. G^+ is triconnected. This implies that $|E^+| \geq 6$, $|V^+| \geq 4$. We examine three exhaustive, but not necessarily disjoint, subcases.

(a) $|K| = 2$: Let $K = \{u,v\}$. If u and v are not adjacent, there exist at least four edges $e^+ \in E^+$ which correspond directly to edges $e \in E$ as opposed to corresponding to chains. Since G^+ is triconnected and contains no parallel edges, $G^+ * e^+$ and $G^+ - e^+$ must both be (at least) biconnected and contain no self-loops. $G^+ * e^+$ and $G^+ - e^+$ may be obtained via series replacements from $G * e$ and $G - e$, respectively, and thus, by Properties 1(b) and 1(c), $\mu(G * e) = \mu(G^+ * e^+) > 0$ and $\mu(G - e) = \mu(G^+ - e^+) > 0$. $\mathbf{S}_2$ is satisfied by e since, by property 1(c), $G * e$ and $G - e$ are biconnected. $\mathbf{S}_1$ is satisfied by e since u and v are not adjacent.

If u and v are adjacent, there exist at least five edges $e^+ \in E^+$, $e^+ \neq (u,v)$, which correspond directly to edges $e \in E$. By the arguments above, any such edge e will satisfy $\mathbf{S}_2$. $\mathbf{S}_1$ is satisifed by e since $e \neq (u,v)$.

(b) $3 \leq |K| \leq 5$: $\mathbf{S}_1$ is satisfied by any edge e since $|K| \geq 3$. Since $|E^+| \geq 6$, there can be at most five chains containing degree-two K-vertices in G_K. Thus, there exists at least one edge $e^+ \in E^+$ which corresponds directly to an edge $e \in E$. By the arguments of case 1(a), $\mathbf{S}_2$ is satisfied by any such edge e.

(c) $|V| - 2 \leq |K| \leq |V|$ and $|K| \geq 3$. $\mathbf{S}_1$ is satisfied by any edge e since $|K| \geq 3$. Thus, as in case 1(a), we need only find an edge $e^+ \in E^+$ which corresponds directly to an edge $e \in E$. It will suffice to find $e^+ = (u,v)$, where $u,v \in K^+$. Such an edge cannot correspond to a chain of length two or three in G, since otherwise G_K would admit a degree-two or series reduction, contrary to assumption. Select any two vertices $x,y \in K^+$. By Menger's well-known theorem, there exist at least three node disjoint paths from x to y since G^+ is triconnected. At least one such path must contain only K-vertices since there are at most two vertices not in K^+. Any edge $e^+ = (u,v)$ in such a path will have $u,v \in K^+$. Therefore $e^+ \in E^+$ corresponds directly to an edge $e \in E$. The rest of the argument follows as in Case 1(a), and thus, any such edge e satisfies $\mathbf{S}_2$.

Case 2. G^+ is biconnected but not triconnected. We examine two subcases.

(a) $2 \leq |K| \leq 5$: By Lemma 1, there are at least two triconnected split components of G^+ with a total of at least ten real edges. At least five of these edges must correspond directly to edges $e \in E$, so select any such edge $e^+ \in E^+$. By Lemma 2, $G^+ * e^+$ and $G^+ - e^+$ are both biconnected and contain no self-loops. By the argument of case 1(a), $\mathbf{S}_2$ is satisfied by edge $e \in E$ corresponding to $e^+ \in E^+$. $\mathbf{S}_1$ will be satisifed by any such e as long as $|K| > 2$ or $|K| = 2$ and the two K-vertices are not adjacent. Suppose $K = \{u,v\}$ and $e = (u,v)$ exists and satisfies $\mathbf{S}_2$. The extra degrees of freedom in choosing an edge which satisfies $\mathbf{S}_2$ allow another edge $e \neq (u,v)$ to be chosen which will also satisfy $\mathbf{S}_1$. (Actually, there are at least nine edges from which to choose when $|K| = 2$ and the K-vertices are adjacent.)

(b) $|V| - 2 \leq |K| \leq |V|$ and $|K| \geq 3$: $\mathbf{S}_1$ will be satisfied by any edge $e \in E$ since $|K| \geq 3$. As in Case 1(b), we need only show that there exists an edge $e = (u,v)$ in one of the triconnected split components of G^+ such that $u,v \in K^+$. There are at least four vertices in two or more triconnected split components of G^+ that are not contained in any separating pair. At least two of these vertices must belong to K^+. Let $u \in K^+$

be any such vertex. Since $\deg(u) \geq 3$ and $|V^+ - K^+| \leq 2$, there exists an edge $e^+ = (u,v) \in E^+$ such that $v \in K^+$. This edge is contained in one of the triconnected split components of G^+ and corresponds directly to an edge $e \in E$. By the arguments of case 2(a), $\mathbf{S}_2$ is satisfied by any such edge e. $\mathbf{S}_1$ will also be satisfied since $|K| > 2$. ∎

Theorem 3 establishes that $\mathbf{S}_1 \cap \mathbf{S}_2$ is achievable and it is the optimal strategy among all possible strategies in $\mathbf{S}_1$. This is the generalization of Chang [5] we were seeking.

Theorem 3. Let G_K be a biconnected graph with no self-loops and with $2 \leq |K| \leq 5$ or $|V| - 2 \leq |K| \leq |V|$. Then, $\mathbf{S} = \mathbf{S}_1 \cap \mathbf{S}_2$ is achievable and optimal for Algorithm 1 under $\mathbf{R} = \{R1,R2,R3,R4\}$ and $\mathbf{S} \subseteq \mathbf{S}_1$.

Proof. If $2 \leq |K| \leq 5$ initially, $|K|$ remains within this range in the graphs created as the algorithm proceeds, because neither factoring nor any reductions ever create additional K-vertices, and because we do not allow factoring on edges so as to create graphs with $|K| = 1$. Similarly, no reductions or factoring can ever create more non-K-vertices, so, if $|V| - 2 \leq |K| \leq |V|$ in the original graph, $|K|$ will remain in this range in all of the graphs encountered as the algorithm factors and reduces.

Since $|K|$ always remains in the given range, and since after factoring we do all possible $R1$–$R4$ reductions, by Theorem 2, the algorithm will always be able to find an edge e satisfying $\mathbf{S}_1 \cap \mathbf{S}_2$. Theorem 1 and its corollary are now sufficient to prove that $\mathbf{S} = \mathbf{S}_1 \cap \mathbf{S}_2$ is optimal for Algorithm 1 under the specified conditions and that $L(G_K) = \mu(G)$. ∎

Complexity of Algorithm 1 under $\mathbf{S}_2$

We discuss next how the complexity of the factoring algorithm changes when we only require that the edge-selection strategy belong to $\mathbf{S}_2$. In an optimization problem, the value of the objective function can only remain the same or improve if a constraint is removed. Analogously, the number of leaves in the backtrack structure of the Algorithm 1 with edge-selection strategy restricted only to $\mathbf{S}_2$ will remain the same or be reduced with respect to the strategy $\mathbf{S}_1 \cap \mathbf{S}_2$.

Theorem 4. Let G_K be a biconnected graph with no self-loops and with $2 \leq |K| \leq 5$ or $|V| - 2 \leq |K| \leq |V|$. Then, Algorithm 1 under $\mathbf{R} = \{R1,R2,R3,R4\}$ and $\mathbf{S} = \mathbf{S}_2$, will give $L(G_K) = \mu(G)$ for $|V| - 2 \leq |K| \leq |V|$, and $L(G_K) \leq \mu(G)$ for $2 \leq |K| \leq 5$.

Proof. Since $\mathbf{S}_1 \cap \mathbf{S}_2$ is always achievable, $\mathbf{S}_2$ is always achievable. We consider two cases.

Case 1. $|V| - 2 \leq |K| \leq |V|$: Consider Algorithm 1 under $\mathbf{S}_1 \cap \mathbf{S}_2$ versus Algorithm 1 under $\mathbf{S}_2$. The algorithms will differ only when $|K'| = 2$ in some induced graph $G'_{K'}$. Let $G'_{K'}$ be such an induced graph. If $|K| = |V|$ or $|K| = |V| - 1$ in the original graph G_K, then $|V'| = 2$ or $|V'| = 3$ in $G'_{K'}$. Thus, $\mu(G') = 1$ and there is no difference in the two variants of the algorithm since $G'_{K'}$ represents a leaf in the backtrack search structure for either variant.

If $|K| = |V| - 2$, however, it is possible that $|K'| = 2$, $|V'| = 4$ and $\mu(G') > 1$. But, this only occurs if G' is the complete graph on four vertices in which case $\mu(G') = 2$. Regardless of which edge is chosen for factoring, the number of leaves below G' will be two. The two variants of Algorithm 1 might not yield the same leaf graphs in their search structures, but they must yield exactly the same number of leaves, namely $\mu(G)$.

Case 2. $2 \leq |K| \leq 5$: Consider the backtrack search structure created by Algorithm 1 under $\mathbf{S}_2$ and all its leaf graphs L. L will consist of two disjoint sets; L_1, those graphs which are single edges; and L_2, all those graphs which are biconnected and which have $|K| = 1$. As in the proof of Theorem 1, we have $\mu(G) = \Sigma_{G' \in L} \mu(G')$. Now, $\mu(G') = 1$ for $G' \in L_1$, but $\mu(G') \geq 1$ for $G' \in L_2$. Therefore, $\mu(G) \geq \Sigma_{G' \in L} 1 = |L| = L(G_K)$. ∎

So, we see that $\mu(G)$ provides an exact value for $L(G_K)$ when $|V| - 2 \leq |K| \leq |V|$ but only an upper bound on $L(G_K)$ when $2 \leq |K| \leq 5$. To see that $\mu(G)$ is a tight upper bound in this latter case, consider Figure 2. Note that there will be no difference in Algorithm 1 under $\mathbf{S}_2$ versus Algorithm 1 under $\mathbf{S}_1 \cap \mathbf{S}_2$; $L(G_K) = \mu(G) = 2$ in both cases. In contrast, $L(G_K)$ can be reduced by an exponential factor by using $\mathbf{S}_2$ in preference to $\mathbf{S}_1 \cap \mathbf{S}_2$. Consider the graph of Figure 2 again, but with the addition of edge $e = (s,t)$. Algorithm 1 under $\mathbf{S}_2$ will produce $L((G + e)_K) = 3$ if e is first chosen for factoring. Algorithm 1 under $\mathbf{S}_1 \cap \mathbf{S}_2$ produces $L((G + e)_K) = \mu(G + e) = 3 \times 2^{(|E|-7)/3}$.

7. CONCLUSIONS

We have investigated the computation of K-terminal reliability by a factoring algorithm which employs the new polygon-to-chain reductions in addition to standard reductions: Algorithm 1 under $\mathbf{R} = \{R1,R2,R3,R4\}$ and under $\mathbf{S} = \mathbf{S}_1$, $\mathbf{S} = \mathbf{S}_1 \cap \mathbf{S}_2$ and $\mathbf{S} = \mathbf{S}_2$. Minimum domination $\mu(G)$ bounds the number of leaves $L(G_K)$ produced by the factoring algorithm when $2 \leq |K| \leq 5$ or $|V| - 2 \leq |K| \leq |V|$: $\mu(G) \leq L(G_K)$ when $\mathbf{S} = \mathbf{S}_1$, $\mu(G) = L(G_K)$ when $\mathbf{S} = \mathbf{S}_1 \cap \mathbf{S}_2$, and $\mu(G) \geq L(G_K)$ when $\mathbf{S} = \mathbf{S}_2$. The range $|K|$ for which the results are valid is limited but includes the most common value of $|K|$, $|K| = 2$. Exponential reductions in computational requirements can be gained over previously studied algorithms which used only $\mathbf{R} = \{R1,R2\}$. In contrast to earlier results which indicated that factoring to produce $|K| = 1$ should be avoided, our results show that, under $\mathbf{S}_2$, factoring in this way will not increase $L(G_K)$ and may significantly reduce it.

The next step in devising even better factoring algorithms for the K-terminal reliability problem may come from the introduction of new reliability-preserving reductions and from edge-selection strategies even less restrictive than $\mathbf{S}_2$. For example, instead of factoring on an edge $e_i = (u,v)$ when $K = \{u,v\}$, an *extended reliability-preserving reduction* can be defined by $G'_{K'} = G_K - e_i$, $\Omega_1 = p_i$ and $\Omega_2 = q_i$ such that $R(G_K) = \Omega_1 + \Omega_2 R(G'_{K'})$. We call this particular extended reduction the *trivial reduction* for obvious reasons. All $R1$–$R4$ reductions are extended reliability-preserving reductions with $\Omega_1 = 0$ and $\Omega_2 = \Omega$, and Algorithm 1 can be easily modified to handle extended reductions.

Consider the graph $G'_{K'}$ intermediate in Algorithm 1. If the algorithm under S_2 is modified to apply the trivial reduction to e in $G'_{K'}$ only when $G' - e$ is biconnected, it follows from Properties 1(a) and 1(c) that $\mu(G' - e) \leq \mu(G')$. Again, as in Theorem 4, $\mu(G)$ will be a tight upper bound on $L(G_K)$ for the original graph G_K. However, it may be advantageous to apply the trivial reduction even when the resulting graph is separable. (This is analogous to factoring on an edge such that $G*e$ is biconnected but $G - e$ is not.) In order to handle efficiently the resulting separable graph, biconnected decomposition should be applied first and the reliability of the individual components computed separately. The straightforward techniques of this paper are not then directly applicable since the backtrack search structure is no longer binary.

References

[1] M. O. Ball, Complexity of network reliability computations. *Networks* **10** (1980) 153–165.

[2] M. O. Ball and J. S. Provan, The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Computing* **12** (1983) 777–788.

[3] R. E. Barlow and F. Proschan, *Statistical Theory of Reliability*. Holt, Rinehart and Winston, New York (1975).

[4] J. A. Buzacott, A recursive algorithm for finding reliability measures related to the connection of nodes in a graph. *Networks* **10** (1980) 311–327.

[5] M. K. Chang, A graph theoretic appraisal of the complexity of network reliability algorithms, Dept. of IEOR, University of California, Berkeley (1981).

[6] L. Fratta and U. Montanari, A Boolean algebra method for computing the terminal reliability in a communication network. *IEEE Trans. Circuit Theory* **CT-20** (1976) 203–211.

[7] F. Harary, *Graph Theory*. Addison-Wesley, Reading, MA (1969).

[8] J. E. Hopcroft and R. E. Tarjan, Dividing a graph into triconnected components. *SIAM J. Computing* **2** (1973) 135–158.

[9] C. L. Hwang, F. A. Tillman, and M. H. Lee, System reliability evaluation techniques for complex/large systems—a review. *IEEE Trans. Reliability* **R-30** (1981) 416–423.

[10] R. Johnson, Some combinatorial aspects of network reliability, Ph.D. Thesis, Dept. of IEOR, University of California, Berkeley (1982).

[11] Moore and Shannon, Reliable circuits using less reliable relays. *J. Franklin Insti.* **262** (1956) 191–208, 281–297.

[12] F. Moskowitz, The analysis of redundancy networks. *AIEE Trans. (Commun. Electron)* **39** (1958) 627–632.

[13] A. Rosenthal, Computing reliability of complex systems, Ph.D. Thesis, University of California, Berkeley (1974).

[14] A. Rosenthal, Computing the reliability of complex networks. *SIAM J. Appli. Math.* **32** (1977) 384–393.

[15] A. Satyanarayana, A unified formula for analysis of some network reliability problems. *IEEE Trans. Reliability* **R-31** (1982) 23–32.

[16] A. Satyanarayana and M. K. Chang, Network reliability and the factoring theorem. *Networks* **13** (1983) 107–120.

[17] A. Satyanarayana and A. Prabhakar, New topological formula and rapid algorithm for reliability analysis of complex systems. *IEEE Trans. Reliability* **R-27** (1978) 82–100.

[18] A. Satyanarayana and R. K. Wood, Polygon-to-chain reductions and network reliability, ORC 82-4, Operations Research Center, University of California, Berkeley (1982).

[19] R. E. Tarjan, Depth-first search and linear graph algorithms. *SIAM J. Comput.* **1** (1972) 146–160.

[20] W. T. Tutte, *Connectivity in Graphs*. University of Toronto Press, Toronto (1980).

[21] L. G. Valiant, The complexity of enumeration and reliability problems. *SIAM J. Comput.* **8** (1979) 410–421.
[22] R. K. Wood, Polygon-to-chain reductions and extensions for computing K-terminal reliability in an undirected network, ORC 81-12, Operations Research Center, University of California, Berkeley (1982).

Received January 25, 1984
Accepted June 14, 1984